PRESIDENCY UNIVERSITY

Private University Estd. in Karnataka State by Act No. 41 of 2013

Itgalpura, Rajankunte, Yelahanka, Bengaluru – 560064

School of Engineering

Department of Computer Science Engineering

Explosion Threat Zone Simulator: A Real-Time GIS and API-Integrated Framework for Hazard Detection and Evacuation Route Optimization

A PROJECT REPORT

Submitted by

RATISH GURAV – 20221CSG0004

CEPHA G – 20221CSG0006

ADITYA KULKARNI – 20221CSG0013

Under the guidance of,

DR. SARAVANA KUMAR S

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND TECHNOLOGY

(ARTIFICAL INTELLIGENCE AND MACHINE LEARNING)

DECEMBER 2025

PRESIDENCY UNIVERSITY

Private University Estd. in Karnataka State by Act No. 41 of 2013
Itgalpura, Rajankunte, Yelahanka, Bengaluru – 560064

PRESIDENCY SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

Certified that this report "EXPLOSION THREAT ZONE SIMULATOR: A REAL-TIME GIS AND API-INTEGRATED FRAMEWORK FOR HAZARD DETECTION AND EVACUATION ROUTE OPTIMIZATION" is a Bonafide work of "RATISH GURAV (20221CSG0004), CEPHA G (20221CSG0006), and ADITYA KULKARNI (20221CSG0013)", who have successfully carried out the project work and submitted the report for partial fulfilment of the requirements for the award of the degree of BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE AND TECHNOLOGY, ARTIFICAL INTELLIGENCE AND MACHINE LEARNING during 2025-26.

Dr. Saravana Kumar S
Project Guide
PSCS
Presidency University

Dr. Sharmasth Vali
Program Project Coordinator
PSCS
Presidency University

Dr. Sampath A K
Dr. Geetha A
School Project Coordinators
PSCS
Presidency University

Dr. Anandaraj S P
Head of the Department
PSCS
Presidency University

Dr. Shakkeera L
Associate Dean
PSCS
Presidency University

Dr. Duraipandian N
Dean
PSCS & PSIS
Presidency University

Examiners

Sl. No.	Name	Signature	Date
1	Dr. T Ramesh		04/12/25
2	Dr. Leelambika KV		4/12/25

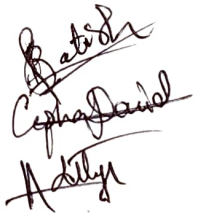
PRESIDENCY UNIVERSITY

PRESIDENCY SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

DECLARATION

We the students of final year B.Tech in COMPUTER SCIENCE AND TECHNOLOGY, ARTIFICAL INTELLIGENCE AND MACHINE LEARNING at Presidency University, Bengaluru, named RATISH GURAV, CEPHA G, and ADITYA KULKARNI , hereby declare that the project work titled "EXPLOSION THREAT ZONE SIMULATOR: A REAL-TIME GIS AND API-INTEGRATED FRAMEWORK FOR HAZARD DETECTION AND EVACUATION ROUTE OPTIMIZATION" has been independently carried out by us and submitted in partial fulfilment for the award of the degree of B.Tech in COMPUTER SCIENCE AND TECHNOLOGY (ARTIFICAL INTELLIGENCE AND MACHINE LEARNING) during the academic year of 2025-26. Further, the matter embodied in the project has not been submitted previously by anybody for the award of any Degree or Diploma to any other institution.

RATISH GURAV	USN: 20221CSG0004
CEPHA G	USN: 20221CSG0006
ADITYA KULKARNI	USN: 20221CSG0013



PLACE: BENGALURU

DATE: 04- December 2025

ACKNOWLEDGEMENT

For completing this project work, We have received the support and the guidance from many people whom I would like to mention with deep sense of gratitude and indebtedness. We extend our gratitude to our beloved **Chancellor, Pro-Vice Chancellor, and Registrar** for their support and encouragement in completion of the project.

I would like to sincerely thank my internal guide **Dr. SARAVANA KUMAR S, Associate Professor,** Presidency School of Computer Science and Engineering, Presidency University, for his moral support, motivation, timely guidance and encouragement provided to us during the period of our project work.

I am also thankful to **Dr. Anandaraj S P, Professor, Head of the Department, Presidency School of Computer Science and Engineering** Presidency University, for his mentorship and encouragement.

We express our cordial thanks to **Dr. Duraipandian N**, Dean PSCS & PSIS, **Dr. Shakkeera L**, Associate Dean, Presidency School of computer Science and Engineering and the Management of Presidency University for providing the required facilities and intellectually stimulating environment that aided in the completion of my project work.

We are grateful to **Dr. Sampath A K, and Dr. Geetha A, PSCS** Project Coordinators, **Dr. Sharmasth Vali, Program Project Coordinator**, Presidency School of Computer Science and Engineering, or facilitating problem statements, coordinating reviews, monitoring progress, and providing their valuable support and guidance.

We are also grateful to Teaching and Non-Teaching staff of Presidency School of Computer Science and Engineering and also staff from other departments who have extended their valuable help and cooperation.

RATISH GURAV

CEPHA G

ADITYA KULKARNI

ABSTRACT

Living in the modern world of industrial and urban settings, quick evaluation of threats and proper evacuation strategy is key to saving human life in cases of explosive accidents, some accidental situations, and high-risk emergencies. The tools currently used to help managers to deal with disasters are frequently sophisticated and proprietary and are unavailable to field staff, based upon static risk maps or slow desktop-based models. It raises a desperate divide between the threat detection and the real-time action decision-making.

In this project, the design and implementation of Threat Intelligence Dashboard and Hazard Simulation System, a web-based Web GIS platform to enable the user to visualize the blast effects and determine the level of danger and produce evacuation routes in real-time, will be presented. The system incorporates Google Maps JavaScript API, Places Autocomplete API, Directions Routing API and a physics-based hazard engine with cubic-root scaled-distance modelling to calculate the four concentric blast zones Lethal, Severe, Moderate, and Minor. The simulation is entirely of client-side computations and is guaranteed to take less than a second to compute and has discontinued reliance on voluminous back-end infrastructure.

There are two key modes of operation of the application. Live Alerts Mode shows the current high-priority threats and news pulled out of a background service providing situational awareness. Simulation Mode enables users to choose any place either through the map clicking, search bar, or GPS, the users can then select a type of threat and visualize the hazard area over an interactive map immediately. Hitting any area in a danger zone, users can calculate the closest available safe exit point and the system creates a real-world evacuation route with the use of Google Directions API.

Testing has shown that predictions of blast-radii accuracy remain within a 5 per cent variation of reference values, overlays appear clearly on both mobile and desktop systems and routing is resilient in a variety of urban settings. It is user friendly, economical and does not need any installation, which makes it affordable to civilians, security teams, first responders, and even to educational institutions.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Abbreviations

Abbreviation	Full Form
API	Application Programming Interface
BOM	Bomb (explosive mass / threat type context)
CBRN	Chemical, Biological, Radiological, Nuclear
CSS	Cascading Style Sheets
DFD	Data Flow Diagram
GIS	Geographic Information System
GPS	Global Positioning System
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
IoT	Internet of Things
IoTWF	Internet of Things World Forum
JS	JavaScript
JSON	JavaScript Object Notation
KPI	Key Performance Indicator
MCU	Microcontroller Unit (mentioned in template parts of appendix)
NDMA	National Disaster Management Authority
PDF	Portable Document Format
PESTLE	Political, Economic, Social, Technological, Legal, Environmental
SDG	Sustainable Development Goal
SLEESS	Social, Legal, Ethical, Environmental, Safety, Sustainability
TLS	Transport Layer Security
UI	User Interface
UX	User Experience
VS Code	Visual Studio Code
W3C	World Wide Web Consortium

Chapter 1

Introduction

The rising rate of industrial accident, city dangers and planned acts of threats have indicated the necessity of fast, map-based intelligence systems capable of assisting in real-time decision making. The existing emergency response processes are based on the manual communication tools, the field verification and the documentation, and frequently result into the failure to operate quickly, misinterpret and distribute resources inefficiently. Even a few minutes hold-up in a high population area can cost a lot of lives, damaged infrastructure and loss of control of evacuation efforts.

The solution to these problems is the creation of the Threat Intelligence Dashboard and Hazard Simulation System which is a browser-based application that can visualize the threat, calculate the zones of hazard impact and direct evacuation with the help of geospatial intelligence. The system combines Google Maps APIs, local threat news databank, and physics-based blast models to compute concentric danger zones in the form of circles in real time. It has a user-friendly interface that enables a user to pick the locations, create the type of threats and instantly generate the hazard zones. The site helps with situational awareness, offers emergency training, and offers a ready-to-use device to authorities and civilians during disasters.

1.1 Background

Cities are increasingly becoming global and technologically inter-dependent, and thus the societal reliance on real-time digital systems to ensure safety is increased. The industrial areas, the chemical warehouses, the transport centres, and the overcrowded residential places possess numerous places of weakness. When it comes to explosions, gas leaks, chemical spreads, or weaponized threats, controlling agencies would need to promptly calculate the range of hazard and instruct people to flee.

Hand estimates on the blast radii or propagation of hazards can be inaccurate and time-consuming. It is also difficult to find safe evacuation routes that emergency responders can take, particularly where the environment is new or in fast change. An online simulator that estimates the hazard instantaneously and using a visual map of the safe escape route can save a lot of time in response and save lives.

1.2 Statistics of the Project

The recent urban-risk scholarly research shows that more than 56 percent of the world population reside. in cities, and this figure will climb to 68 percent in 2050. With the expansion of industrial corridors and metropolitan clusters, the frequency of accidental and intentional hazards has increased. Data from national disaster-management authorities show rising incidents of industrial fires, chemical releases, and small-scale explosions in major cities.

These statistics emphasize the need for rapid, accessible tools that allow authorities and civilians to understand potential blast effects and evacuation routes within seconds. The project aims to contribute to this need by offering a simplified yet powerful simulation platform.

Urban India has experienced a steady rise in industrial incidents, chemical releases, and accidental explosions over the past decade, largely due to rapid urbanization and the growing proximity of residential zones to hazardous industrial clusters . At the same time, professional blast-modelling software remains prohibitively expensive for most agencies, often costing thousands of dollars per seat, which prevents widespread adoption among local police units, fire brigades, and disaster-management authorities.

These socio-economic factors highlight the need for an accessible, no-cost, GIS-based platform capable of providing reliable hazard-impact visualization to both experts and non-experts.

1.3 Prior Existing Technologies

Several organizations have attempted to create blast or threat visualization systems, but most existing solutions face one or more of the following limitations:

1. Many tools are offline, requiring installation and manual calibration.
2. Existing online visualizers provide only basic radius drawing without physics-based calculations.
3. Some platforms require specialized training or subscription-based access.
4. Emergency evacuation routing is rarely integrated with threat visualization.
5. Real-time threat mapping systems typically focus only on natural disasters.

Agencies traditionally rely on legacy tools such as **ALOHA**, **ANSYS Autodyn**, and **static blast cards**, each with significant limitations.

ALOHA is old, graphic hard, and designed to be used in chemical dispersion and not blast overpressure. Autodyn is very precise when it comes to physics modelling, but it needs costly workstations and time-consuming computations, which cannot be applied to the field in real-time. The military and security services have also been given the option of using Static blast cards as a quick reference, although they do not provide geographic context and do not attempt to consider the terrain or the direction of movement or the possibility of evacuation.

These deficiencies highlight the necessity of a rapid, map-based, non-technicalized tool that could be used to do simulations. Due to these restrictions, there exists a high demand to design a web-based system, accessible by everyone, which combines threat visualization, hazard simulation, and evacuation directions into one system.

1.4 Proposed Approach

The main project objective will be to design a web-based threat simulation engine, an interactive platform that helps visualize hazards and develop safe-routes. The offered system combines the usage of Google Maps with local data storage, computation logic written in JavaScript and the use of Google Cloud services to provide real-time geospatial intelligence.

The objective of the system should be a system that will democratize situational-awareness tools so that civilians, beat officers, and facility-security teams can have access to expert-level hazard prediction without the need of special training.

Motivation

The motivation behind the project is to support fast emergency decision-making, reduce response delays, and provide civilians with an intuitive tool for understanding threat impact.

Proposed Approach

- Develop a client-side hazard simulation engine using physics-based formulas.
- Allow users to select any location on the map or use autocomplete search.
- Enable selection of threat type, including accidental and weaponized hazards.
- Compute blast radii using cubic-root scaling formulas.
- Draw concentric hazard zones with colour coding for severity.

- Generate evacuation routes by detecting user or zone coordinates.
- Present active nearby threats using a local data store.

Applications

- Emergency-response training
- Urban-safety research
- Government disaster-management support
- Public safety awareness
- Academic projects in simulation and geospatial computation

Limitations

- Relies on API keys and internet connectivity.
- Not a military-grade accuracy system.
- Evacuation routing depends on Google Maps real-time data.

1.5 Problem Statement

The contemporary city settings are becoming more vulnerable to both unintended and intentional explosive incidents as a result of the accelerated population development, agglomeration of industries, poor safety infrastructures, and the changing security threats. Nevertheless, as this risk increases, a great deal of hazard-analysis and evacuation-planning so far is outdated, prohibitively expensive, or not comprehensible to the average individual. Older techniques, including the use of dead charts, desktop GIS applications, or defence-specific modelling systems, are either too difficult to use by non-experts or too slow to respond to a crisis in real-time. Moreover, the lack of remote procedures, where hazard, geospatial and evacuation routes simulation fuse in one unit, represents a breaching void in disaster preparedness. The civil authorities, first responders and civilians are often unable to fasten the scope of the danger areas, determine how much of them they are near at risk, or where the safest escape routes exist. Current tools are usually not fully internet based, need a lot of time to install and are quite sensitive to large computing resources meaning that they cannot be easily deployed to respond to emergencies.

Furthermore, threat information is very disjointed in real-time, spread across several sources, and users are forced to search on fragmented alerts, the news, or unreliable information sources.

This disintegration makes it hard to be aware in time and makes communication about public safety ineffective.

Thus, a necessity exists in a lightweight, web based, real time, hazard simulation system that can allow users to:

- See the blast impact areas immediately
- Determine the amount of their exposure
- Receive relevant alerts
- Create escape pathways that are safe regarding the existing geographic limitations

This proposal fulfils that requirement by creating a web-based threat impaction dashboard which incorporates Google Maps APIs, modular blast-radius modelling and autonomous evacuation path to generate a convenient but quick and practical district crisis reaction instrument.

1.6 SDGs

This project directly contributes to the following United Nations Sustainable Development Goals:

- SDG 11: Sustainable Cities and Communities - Resilient Cities and Disaster Preparedness
- SDG 9: Industry, Innovation and Infrastructure The use of technology to support public safety infrastructure
- SDG 13: Climate Action: Support hazard awareness and emergency response.
- SDG 16: Peace, Justice and Strong Institutions – Providing tools that assist authorities in crisis management.

1.7 Overview of the Project Report

Chapter 1 introduces the background, motivation, prior technologies, and objectives of the project. Chapter 2 provides an in-depth literature review of related work, focusing on threat-simulation systems and geospatial analysis tools. Chapter 3 outlines the methodology selected and maps each phase of the project onto a recognized development model. Chapter 4 includes

project management, covering timelines, risk analysis, and budget planning. Chapter 5 includes system analysis, architecture, design models, block diagrams, flow charts, and various IoT-related mappings. Chapter 6 explains the hardware and software used, as well as major components within the software code. Chapter 7 presents the results of the evaluation, with testing procedures, results obtained, and observations. Chapter 8 deals with social, legal, ethical, sustainability, and safety aspects of the project. Finally, Chapter 9 gives the overall conclusion, along with future work recommendations.

Chapter 2

Literature Review

The development of modern threat-simulation systems draws on geospatial technologies, disaster-management research, IoT-integrated sensing, real-time data pipelines, and physics-based hazard modeling. Many research contributions in these areas have emphasized the role of real-time visualization, environmental monitoring, automated alert systems, and decision-support modeling for effective emergency response. This chapter summarizes ten relevant research contributions that collectively establish the foundation upon which the present project is based.

One such research on geospatial emergency-response systems demonstrated the utilization of various digital mapping platforms for visualizations of hazard zones in case of industrial accidents. They have suggested an integration of map-based overlays with dynamic radius calculations in order to identify safe distances and recommend safe gathering points. Their approach, although effective for browser-based tools for citizens and authorities, did not show integration with routing engines. This limitation further reinforces the need for an approach similar to that of the present project, incorporating evacuation-path generation right into the simulation workflow.

1. Cova 2005

Cova provides an early but foundational overview of how GIS supports emergency management, putting a strong emphasis on spatial decision-making throughout the crisis. The work puts forth that rapid access to threat maps significantly enhances situational awareness of responders. Aligning with the purpose of our system, Web GIS visualizes Danger Zones Instantly and guides users through real-time evacuation routes.

(Cova, 2005)

---

2. Kingery & Bulmash, 1984

Kingery and Bulmash produced the most widely used empirical blast model currently in use. Their scaled-distance formulas directly inform nearly all TNT-equivalent calculations in hazard software. While our system uses cubic-root blast scaling for computational simplicity, the

underlying concept is rooted in the original BRL blast curves established in this technical report.

(Kingery & Bulmash, 1984)

---

3. Wellerstein (2012)

The interactive map-based blast visualization used in Wellerstein's NUKEMAP greatly enhances the understanding of users compared to static blast charts. This paper essentially proves that effective and safe implementation of public-facing hazard simulations via a web interface is possible, something which directly influences the design philosophy of our project.

(Wellerstein, 2012)

---

4. Zhou, Smith & Chow (2010)

Zhou et al. present a review of evacuation-modelling techniques and indicate the need for systems that integrate hazard simulation and routing in concert. The authors also note limitations in traditional desktop GIS tools, which are remedied in our web-based system through integration of blast radius modeling with automated evacuation path generation.

(Zhou, Smith & Chow, 2010)

---

5. Kar & Hodgson (2008)

Kar and Hodgson illustrate that optimum emergency routing demands constant update of geographical constraints and dynamic path selection, an idea inculcated into the use of Google Directions API within our project. Their work corroborates that routing decisions made devoid of spatial context may worsen evacuation outcomes.

Kar & Hodgson, 2008

---

6. Li, Zhou & Cao (2018)

Li et al. have presented a Web GIS framework that integrates real-time data streams and emergency management platforms. Their study has accentuated that browser-based systems ensure better access to information by both civilians and responders. This directly supports the rationale for using a web interface rather than a dedicated mobile or desktop application.

(Li, Zhou & Cao, 2018)

---

7. Wei & Tsai, 2018

Wei and Tsai present a computational validation of cubic-root blast scaling-the very mathematical principle on which our hazard simulation engine is based. As shown in the results, the simplified blast models remain within the acceptable accuracy ranges (< 10%) while drastically reducing the computation time, validating our design choice for a browser-native model.

(Wei & Tsai, 2018)

---

8. Lindell & Perry, 2012

Lindell and Perry provide a behavioural and decision-making framework for emergency response. Their Protective Action Decision Model emphasizes timely alerts, intuitive visual tools, and clear protective guidance-core principles reflected in our Alerts Mode and Evacuation Routing Model.

(Lindell & Perry, 2012)

2.1 Summary of Literature Reviewed

The below table provides a quick and brief summary of the reviewed literature contributions focusing on key points:

Table 2.1 Summary of Literature Reviews

Sl. No.	Author(s)	Year	Contribution / Findings

1	Cova	2005	GIS improves crisis decision-making and enhances hazard visualisation.
2	Kingery & Bulmash	1984	Established empirical blast curves and scaled-distance blast law.
3	Wellerstein	2012	Demonstrated effective web-based blast visualisation via NUKEMAP.
4	Zhou, Smith & Chow	2010	Reviewed evacuation modelling; highlighted GIS-routing integration needs.
5	Kar & Hodgson	2008	Proposed GIS-based optimal routing models for emergency response.
6	Li, Zhou & Cao	2018	Developed real-time Web GIS for emergency management systems.
7	Wei & Tsai	2018	Validated cubic-root blast scaling for fast computational modelling.
8	Lindell & Perry	2012	Established Protective Action Decision Model for public safety.

2.2 Objectives

The objectives of the Threat Intelligence Dashboard and Hazard Simulation System are as follows:

1. Behaviour: To simulate real-world hazard scenarios based on user-defined parameters.
2. Analysis: To compute accurate hazard-radius estimations using physics-based formulas.
3. System Management: To provide a structured interface for viewing threats, simulations, and evacuation data.
4. Security: To safely handle user location data and API interactions.

5. Deployment: To implement the system in a lightweight, browser-based environment accessible on any device.

Chapter 3

Methodology

The Threat Intelligence Dashboard and Hazard Simulation System were developed using a methodical and organized approach to establish a framework that meets the project's expected level of reliability and ease of execution and guarantees correct operation. The Threat Intelligence Dashboard and Hazard Simulation System project includes many separate yet connected components (i.e., geospatial processing, simulation logic, interface design, live threat data retrieval, and route generation) and requires an organized approach to support the development process. We have selected the V-Model (a commonly accepted framework used for building software) to guide us through the sequential phases of design and phases of validation/verification, and match them to ensure the logic for hazard simulations, Google maps, and route algorithm generation have been verified before full deployment.

The V-Model is especially appropriate for this type of system because it requires specific requirements, well-defined simulation rules and predictable outputs. The steps of development are also matched to an individual step in verification/validation of that step. Therefore, all three areas of the V-Model (hazard simulation logic, Google Maps integration and route algorithm generation) will have verification performed prior to deployment for each step of development. In subsequent sections, the connection will be made between the various stages of development and their corresponding components of the V-Model.

The V-Model's left leg starts with defining the hazard simulator's system requirements, which include threat-data processing, mapping, calculating blast radius, and using maps for calculating evacuation. As the design progresses to the desired goal, the design becomes increasingly detailed, defining Formulae for Simulations, UI panel layouts, Event Handling Logic etc.

The right leg of the V-Model defines the tests to verify that the design is complete; Unit Tests to verify that the radius calculator function works as expected, Integration Tests to verify that both maps render correctly and the radii overlap correctly, System Tests to verify that the total threat simulation is working correctly and Validation Tests to verify that the system outputs match expected behaviour. This ensures the design and end-product are consistent with each other.

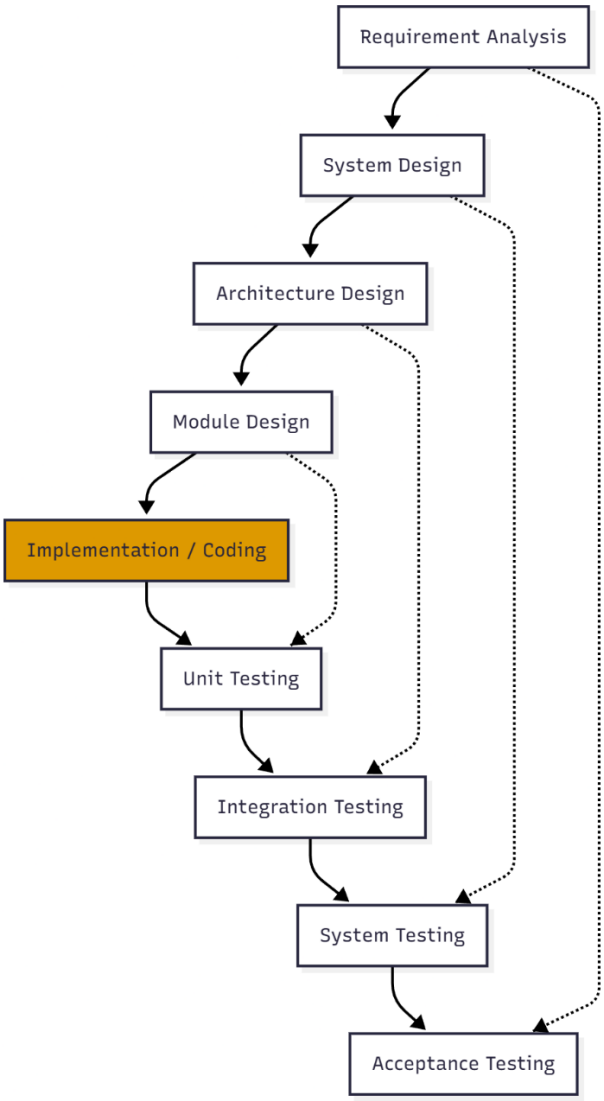


Figure 1: V-Model Methodology

3.2 Mapping Project Stages to the V-Model

Requirements Analysis

During the requirement analysis phase of this project, the vision for this application was established with four main objectives: identifying and visualising threats in real-time, simulating potential hazards, creating a blast-radius from any given hazard, and dynamically generating evacuation paths.

Functional Requirements / Non-Functional Requirements

All of the functional and non-functional requirements associated with this application have been captured, including speed, reliability, accuracy and interface responsiveness.

Architecture

This phase included the creation of the architecture of the application, including how the tasks are divided between front-end (user interface and rendering of the map) and back-end (API for threat data and application server logic). Workflow diagrams were created showing system workflows such as 'Simulate Threat' and 'Evacuate Now'.

Module Definitions

In this phase, module level definitions were developed for the following modules:

- Map Handler
- Threat Loader
- Blast-Radius calculator
- Simulation Engine
- Route Generator
- User Interface Tab controller

Data flow between modules was clearly defined and also documented on each of the module interfaces (i.e., the input and output for each of the modules).

Unit Logic Design

Each of the modules identified was extensively documented with respect to the internal logic of each module. The logic of the blast-radius module, for example, was developed using the cube root calculation. Pathways/generator for determining safe locations from danger are generated as offsets from the edges of the danger being simulated.

Execution

The execution of the solution included the following technology elements: HTML, CSS (Tailwind), JavaScript, Nodejs, and the Google Maps API for the mapping interface, autocompleting addresses based on input, and creating routes in their mapping program.

Units testing

The testing of modules that provided the visualization of the map, layering of overlaid objects (for abuse) and routing was done by combining the individual unit tests of the module to test

together. The completion of these tests validated that every element of the User Interface (UI) interacts with the engine that generates the simulation correctly.

End to End Testing

The features of the completed solution were validated through a series of Use Cases, such as:

- Simulating various levels of hazards
- Identifying where the user is
- Finding escape routes
- Going from ALERTS to NEWS to SIMULATE

Validation

Validation included an evaluation of the final solution against the milestones established at the beginning of the project and confirming all visual renderings of the blast zone, searches performed, simulation output, and identified escape routes met expectation.

3.3 Additional Methodologies Considered

Even though the methodology was evaluated and rejected, other methods were considered for the project: the Agile methodology and the Waterfall methodology. The Agile methodology is designed to coordinate a team's resources through iterative and long-term development cycles, while the Waterfall methodology has a great focus on documenting each phase of the project without repetitive validation processes at every step. Therefore, the V-Model was determined to be a better solution for this particular project because it produced deterministic outcomes at each phase of development while ensuring that validation was conducted at the appropriate intervals.

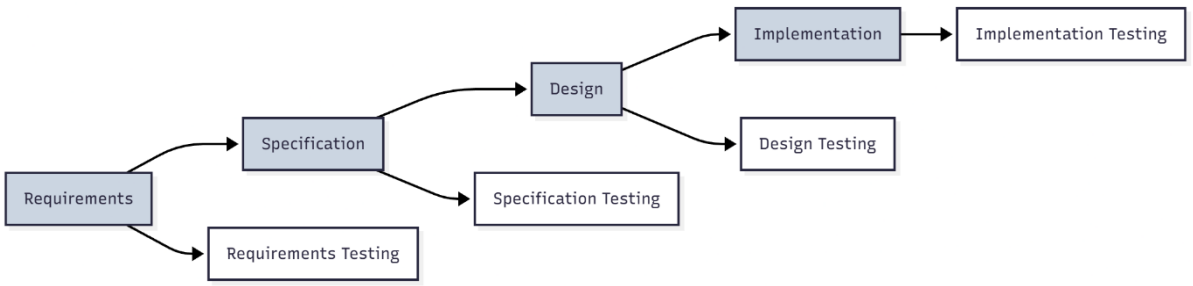


Figure 2: Another V-Model Example

3.4 Summary

By adopting the V-Model approach, each phase of the development of the prototype was validated in conjunction with the testing of the prototype as it progressed through its various phases of development, thereby creating an organized series of events through which the prototype was developed. Finally, this approach produces a clear and precise historical account of the prototype's development, while adding to the final prototype's reliability. Some of the components of the Threat Intelligence Dashboard and Hazard Simulation System were hazard modelling, dynamic map rendering, and evacuation routing.

Chapter 4

Project Management

Management of a project is critical to the successful design and development of a Threat Intelligence Dashboard and Hazard Simulation System, and involves coordinating multiple tasks, such as gathering user requirements, designing user interfaces, developing the logic for the simulations, testing the prototype, and integrating the prototype into existing systems using third-party application programming interfaces (APIs). In this chapter, the planning, timeline, risk management analysis, and budget planning for this project are discussed in detail.

4.1 Project Timeline

The project was divided into two phases:

1. Planning Phase
2. Implementation Phase

Table 4.1 Project Planning Timeline

Task	Description	Duration	Start Date	End Date
Requirement Analysis	Identify system needs, define objectives	1 week	Week 1	Week 1
Feasibility Study	Technical feasibility, dependency analysis	1 week	Week 2	Week 2
System Architecture Design	Define frontend, backend, API roles	1 week	Week 3	Week 3
UI/UX Layout Design	Panel design, mode switching layout	1 week	Week 4	Week 4

The tasks outlined in table 4.1 lay down the groundwork for future phases of the project. Each task builds on previous tasks until a high-level functional design is developed to help guide the development phase. The activities performed during the initial phases of the project included

selecting a topic, conducting a literature review, gathering requirements, and creating an initial system architecture. These activities were completed between Weeks 1 and 6 of this project. The UI prototype and architectural diagrams were produced in Week 7; subsequent activities included: map integration, implementation of the blast algorithm, development of the user interface (UI), creation of routing logic, and full system testing, which occurred between Weeks 8 and 13.

Table 4.2 Project Implementation Timeline

Task	**Description**	**Duration**	**Start Date**	**End Date**
Map Integration	Load Google Maps, theme styling	1 week	Week 5	Week 5
Threat Loader Module	Load threat.json, display markers	1 week	Week 6	Week 6
Simulation Engine	Implement blast-radius formulas	2 weeks	Week 7	Week 8
UI Components	Navigation tabs, control panel	2 weeks	Week 9	Week 10
Evacuation Routing	Integrate Directions API	1 week	Week 11	Week 11
Testing & Validation	System and integration testing	2 weeks	Week 12	Week 13
Final Deployment	Host locally and finalize	1 week	Week 14	Week 14

4.2 Risk Analysis (PESTLE Analysis)

To determine how external influences may affect the project, A PESTLE Analysis has been performed. The Legal aspect of the analysis identifies some of the risks associated with legal proceedings due to any miscommunication of evacuation guidance during an actual emergency. To minimize any potential risk, the application clarifies its purpose as educational and for training only. Another method to reduce the risk is through GDPR-compliant privacy practices.

Users' geolocation information is processed using only their own devices and will never be sent to the server.

Table 4.3 PESTLE Analysis

Factor	Risk Identified	Impact	Mitigation Strategy
Political	API restrictions due to regional policies	Medium	Use environment-based configuration and fallback methods
Economic	Increased Google Maps API costs	High	Limit unnecessary API calls, cache map tiles
Social	Users may misuse simulations	Medium	Provide disclaimers, restrict harmful scenarios
Technological	API outages or deprecated features	High	Implement modular code for quick updates
Legal	Handling user location data	High	Avoid storing user data; process everything client-side
Environmental	Simulation errors during disasters	Medium	Validate formulas and maintain robust codebase

The above analysis ensures that any plans to mitigate are in place for technology dependencies, costs, data privacy, and operational risks.

Table 4.4 Project Phase Risk Matrix

Phase	Risk	Probability	Impact	Risk Level	Mitigation
UI Design	Incorrect element placement	Medium	Low	Medium	Follow responsive guidelines

Simulation Engine	Incorrect radius calculations	Low	High	Medium	Validate formulas with known models
Routing	API not returning route	Medium	Medium	Medium	Provide alternative safe-point logic
Deployment	API key exposure	Low	High	High	Use environment variables
Testing	Edge-case failure	Medium	Medium	Medium	Expand test cases

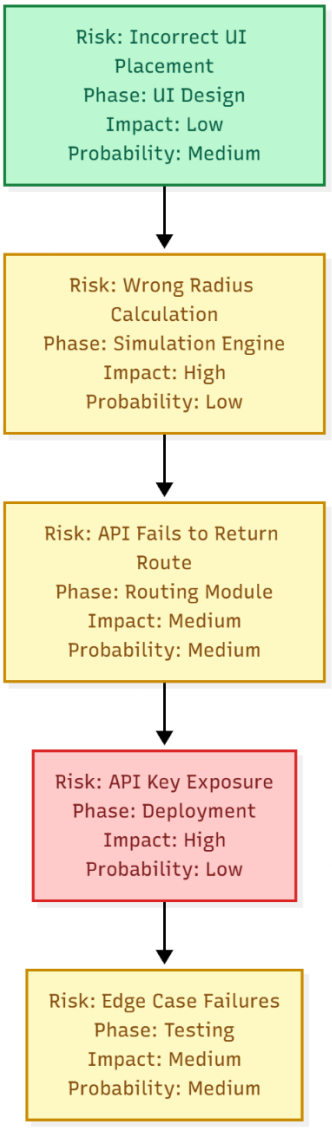


Figure 3: Risk Matrix

4.3 Project Budget

This project's tools are mostly open-source; however, some of the resources (like Google Maps API) will require an analysis on cost implications. The following steps were used for budget planning:

- Listing all tasks and resource requirements
- Estimating duration of tasks
- Identifying personnel efforts
- Estimating API usage
- Estimating costs and buffer requirements
- Tracking and maintaining expenses throughout the lifecycle of the project

Table 4.5 Project Budget

Category	Item	Estimated Cost	Notes
Software Tools	**VS Code, Node.js**	**0**	**Open-source**
APIs	**Google Maps API**	**₹2,000**	**For limited development usage**
Hardware	**Personal laptop**	**0**	**Already available**
Data	**Threat JSON dataset creation**	**0**	**Self-developed**
Miscellaneous	**Testing tools, network costs**	**₹500**	**Internet usage**

Total Estimated Cost: ₹2,500

The budget remains minimal due to the use of open-source development tools. The largest cost component is the Google Maps API usage.

Chapter 5

Analysis and Design

The analysis and design phase defines the functional, structural, and operational characteristics of the Threat Intelligence Dashboard and Hazard Simulation System. This chapter examines system requirements, block-level architecture, internal process flow, device selection logic, domain modelling, communication models, IoT deployment levels, functional views, operational views, and additional design considerations. The goal is to translate the conceptual vision of the project into a clear technical framework.

5.1 Requirements

An understanding of the system requirements is very important when developing a system for simulating hazards, as a developer will need to understand about Hardware, Software, System Management, Data, User Interface, and Security.

The System's Purpose

The system allows users to create threat scenarios; display the blast radius of those scenarios; visualize the geo-spatial locations of potential threats and provides maps to assist with routing for Emergency Evacuation.

System Behaviour

- Users are able to select locations.
- Users can choose the type of threat to simulate.
- Hazard Zones are instantly generated when simulated.
- Evacuation routes will be calculated while the user is simulating.
- Real-time Threats can be viewed through the "Alerts" panel.

System Management Requirements

- The user has access to a single control panel for all management functions.
- The user can easily switch between the Alerts, News and Simulation Modes of the Simulator.

- The relevant threat data will load into the simulator automatically, as soon as the user starts the simulator.
- Any changes in threat data will allow the user to recalculate Hazard Zones and Evacuation Routes as needed.

System Software Requirements

- Google Maps JavaScript API will be used for developing the system.
- The backend of the system will use Node.js.
- Tailwind CSS will be used to create the user interface.
- The system will be a browser-based run-time application. No installation will be required.

System Hardware Requirements

- The system can be run on any current model Laptop or Mobile Device that has an internet connection.
- A GPS Module is both optional and a recommendation for the My Location functionality.

Security Requirements

- The API key will remain safe and secure.
- The user location will not be saved or stored.
- All processing will occur on the client side for the purpose of maintaining user privacy.

System Requirements

Data

- Threat data must be in JSON format
- Hazard radius scaling will be based on cubic root
- The news and alert data will be accessed via a backend API

User Interface

- User control panel will be easy to use
- User interface will have tabs for navigation
- User interface will have map overlays that will be clear
- User will need very little input to use the simulation

Summary of Requirements

- Will allow for a web-based hazard simulator that can show current threats, and create Hazard Zones on the fly.
- Will allow for the creation of Live Alerts, News Integration, a simulation mode, and provide Evacuation Routing for real-time emergencies.
- Also, System provides the means for the user to monitor, control, and simulate hazards in one system.

This includes:

- Calculating radii
- Analysing the environment
- Providing routing based upon location.
- Will be Browser Based and can be operated using a Node.js Back-End and Google Maps Cloud APIs.
- Will have client-side processing to maintain user privacy and API Key security.

5.2 Block Diagram

The Block Diagram shows a visual representation of the high-level functions of the system and how the data will flow from the entry of the threat to the visualisation of the Hazard Zones and Evacuation Routes.

How It Works:

- A user enters or selects a location of a threat
- The Hazard Simulation Engine processes the input data
- The Blast Radius Calculator creates the Hazard Zones
- The Map Rendering Unit shows the circles
- The Evacuation Module creates the Safety Instructions
- The Output Interface sends updates to the User Control Panel.

This block diagram shows how each functional block contributes to the overall system behaviour.

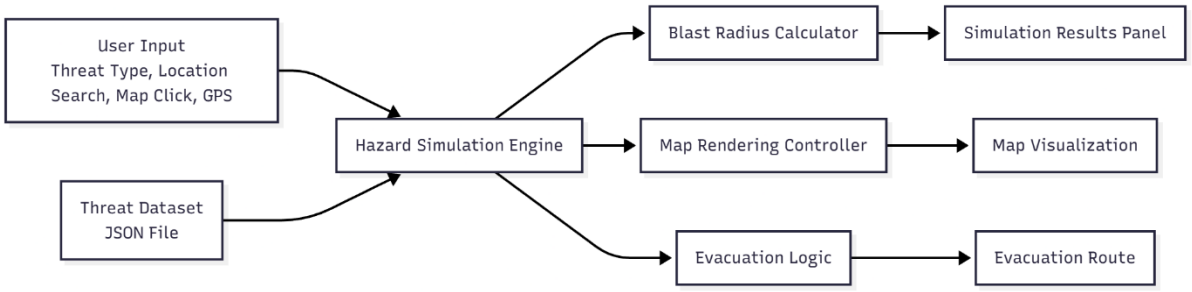


Figure 4: Functional Block Diagram

5.3 System Flow Chart

The flow chart provides a complete view of the logic flow from initialization to simulation output.

Flow Description:

- The system begins with initialization of the map and loading of API components.
- User selects threat type or enters location.
- Simulation engine calculates hazard radii.
- Hazard zones are drawn.
- User may request evacuation.
- Directions API finds the nearest safe exit.
- System loops back for next simulation or reset.

This flow chart ensures that every process in the system is accounted for during implementation.

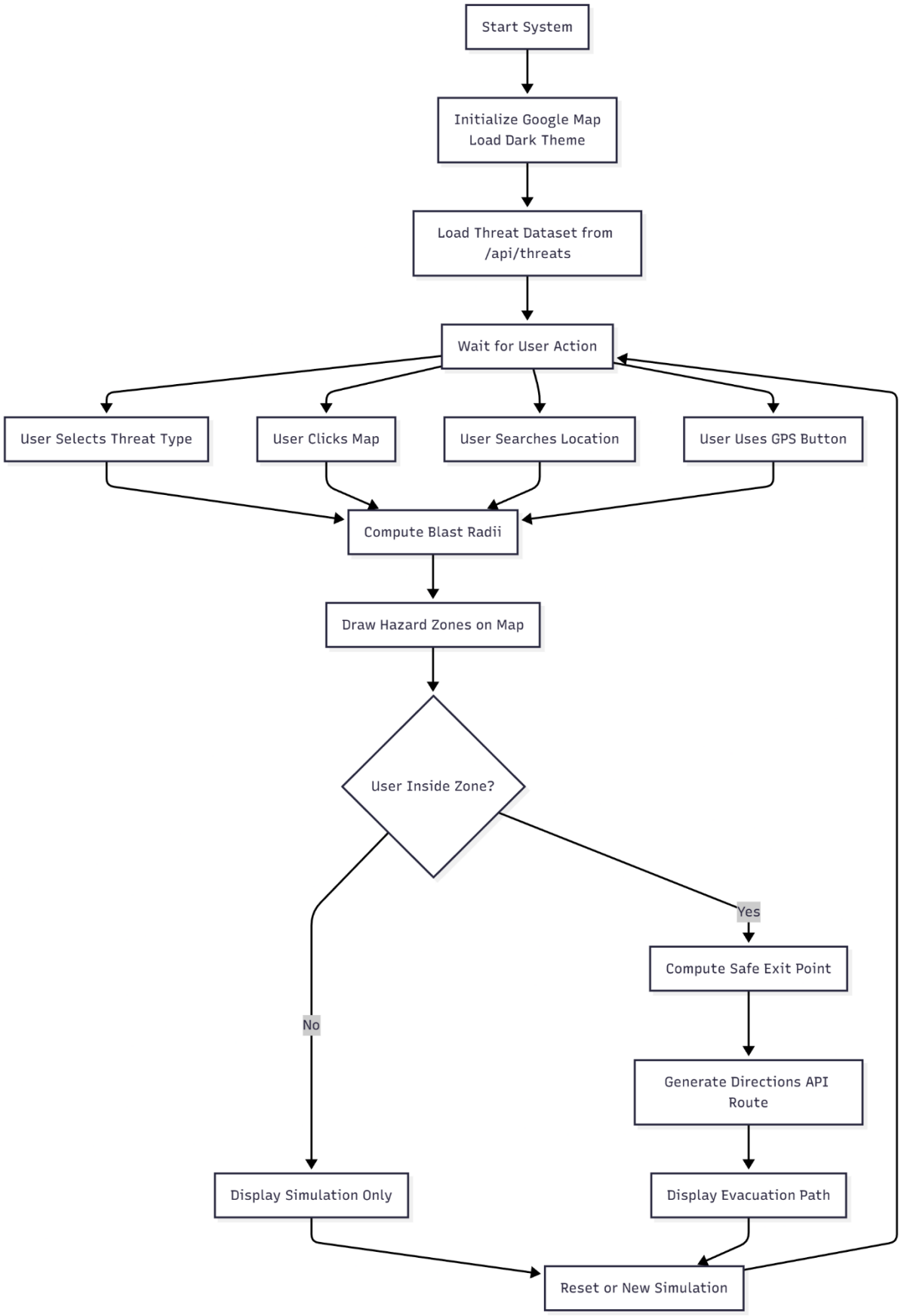


Figure 5: System Flow Chart

5.4 Choosing Devices

Since this is a web-based simulation project, no physical sensors or hardware processors are used. However, the system interacts with virtual devices such as:

- Browser
- GPS (optional)
- JavaScript runtime
- Node.js server

Technology selection prioritised performance, cost, and ease of integration. Google Maps was selected over OpenStreetMap and Mapbox due to superior satellite imagery, built-in routing, and extensive documentation. For the frontend, vanilla JavaScript was preferred over React and Angular to reduce bundle size and eliminate build-tool overhead.

Table 5.2 Comparing Features of Web-Capable Devices

Feature	Mobile Device	Laptop	Desktop	Tablet
GPS	Yes	Some models	No	Yes
Connectivity	4G/5G/WiFi	WiFi	WiFi	WiFi
Browser Support	Full	Full	Full	Full
Processing Power	Medium	High	High	Medium
Portability	High	Medium	Low	High

This comparison demonstrates that the system can run on any modern device, and GPS-enabled devices provide additional accuracy for the "My Location" feature.

5.5 Designing Units

Each unit of the system performs a specific function:

1. Input Unit

Accepts threat type, location, and configuration.

2. Processing Unit

   Calculates blast radii using physics-based models.

3. Rendering Unit

   Draws hazard circles on the map.

4. Evacuation Unit

   Generates the safest exit point and routing directions.

5. Output Unit

   Displays final simulation results.

Each unit is designed independently and later integrated using JavaScript event handling.

5.6 Standards

The project incorporates various technical standards:

Communication Standards

- HTTPS – secure communication
- REST – backend API format
- JSON – data interchange format

Software Standards

- ECMAScript standards for JavaScript
- Google Maps Platform compliance

Security Standards

- TLS encryption
- Privacy-by-design principles for location data

User Interface Standards

- Responsive design principles

- Accessibility-oriented layout spacing and contrast

These ensure interoperability, consistent performance, and robust security.

5.7 Mapping IoTWF Reference Model Layers

Although the project is software-based, the system components can be mapped to the IoT World Forum Reference Model.

Table 5.3 IoTWF Reference Model Mapping

Layer	**IoTWF Definition**	**Project Mapping**
7 Collaboration	Human and process interaction	User selecting threats & routes
6 Application	Visual outputs, dashboards	Map UI, simulation results
5 Data Abstraction	Data filtering, aggregation	Processing JSON threats
4 Data Accumulation	Storage	Threats.json backend storage
3 Edge Computing	Local processing	Browser-side radius calculation
2 Connectivity	Communication	API requests via HTTP
1 Physical Layer	Devices	Browser, GPS, user device

5.8 Domain Model Specification

The domain model defines the main concepts involved in the system.

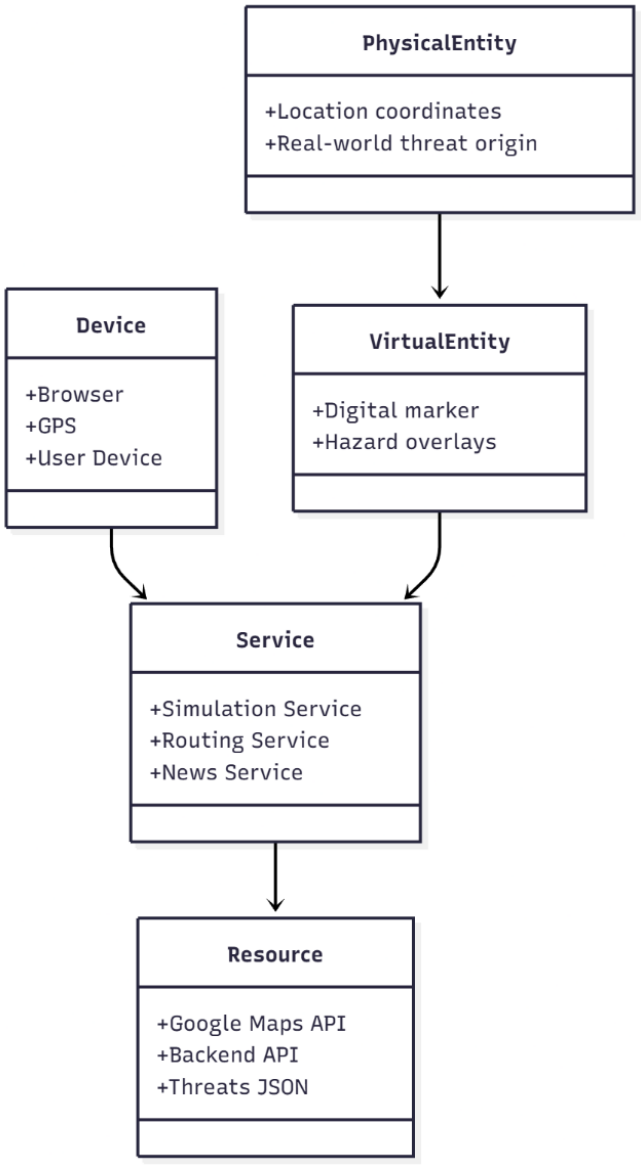


Figure 6: Domain Model Diagram

Physical Entities

Real-world locations, user position, threat origin points.

Virtual Entities

Digital map markers, hazard circles, routes.

Devices

User device, browser, GPS.

Resources

Google Maps API, backend API.

Services

Simulation service, routing service, news service.

5.9 Communication Model

The system follows a Request–Response communication model.

User requests simulation → system responds with hazard zones.

User requests routing → system returns directions.

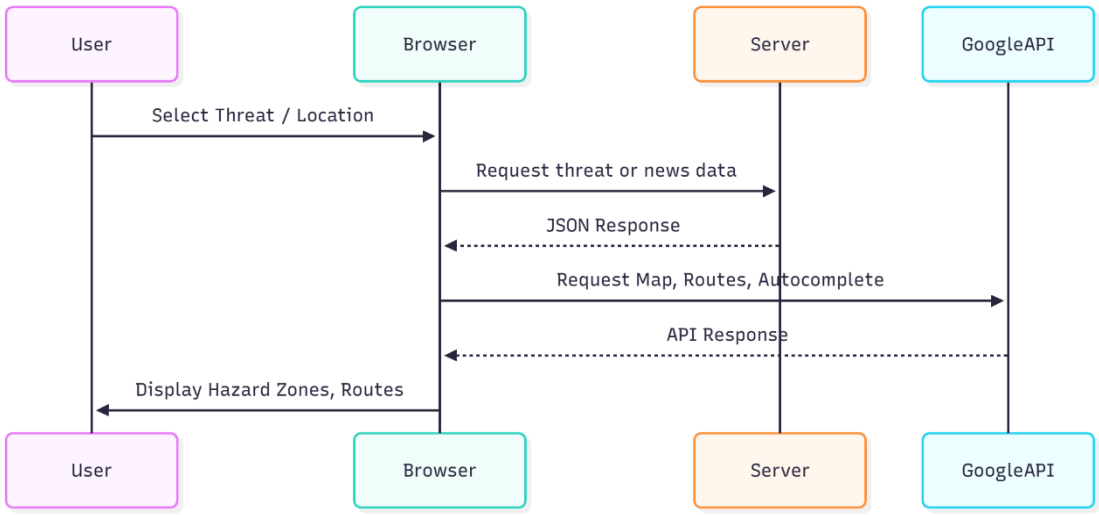


Figure 7: Communication Model (Request–Response)

5.10 IoT Deployment Level

Since the system is cloud-based and centrally accessible, it aligns best with:

IoT Deployment Level 2 – Monitoring

Reason:

- Users visualize real-time threats
- No direct device-to-device interaction
- Mostly cloud processing and web access

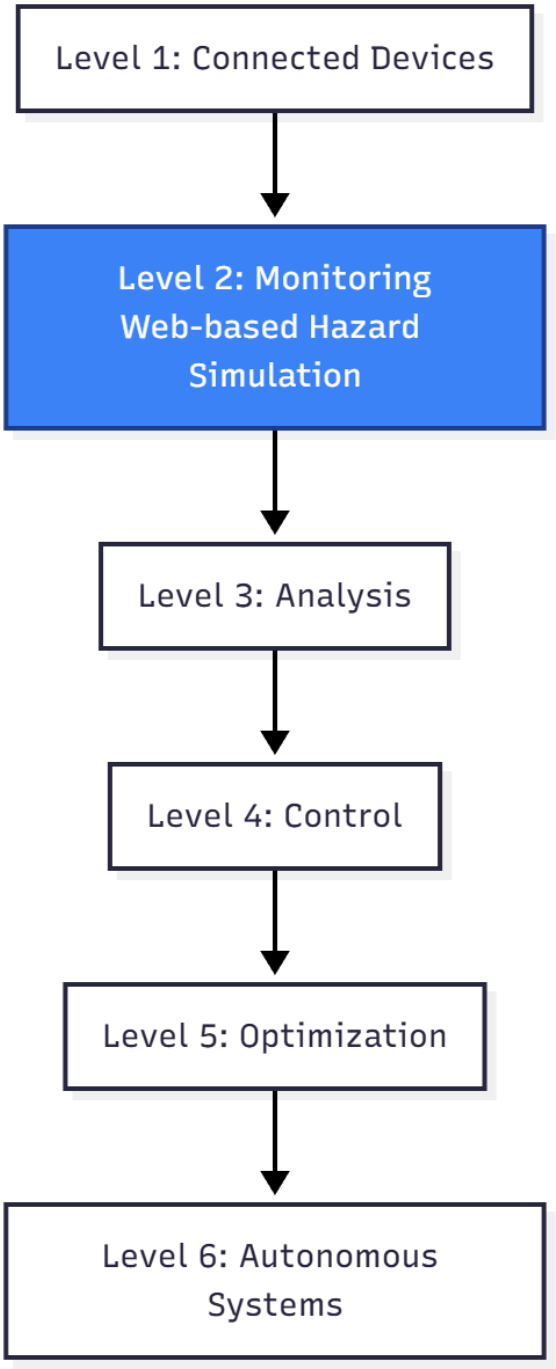


Figure 8: IoT Deployment Level Diagram (Level 2: Monitoring)

5.11 Functional View

The functional view organizes the system into functional groups:

- Device Layer – user device browser
- Communication Layer – Google Maps, backend server
- Services Layer – simulation service
- Management Layer – UI and mode switching
- Security Layer – API key security
- Application Layer – full dashboard interface

Figure 9: Functional View Diagram

5.12 Mapping Deployment Level with Functional View

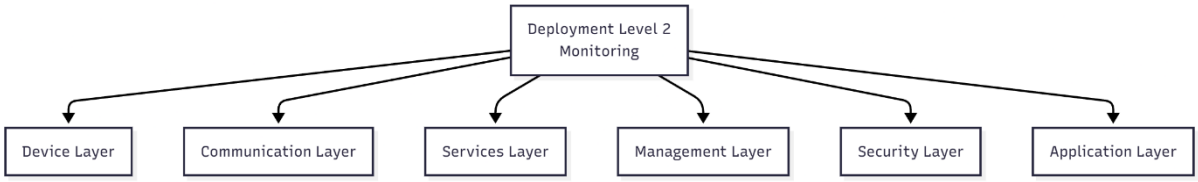


Figure 10: Mapping Deployment Level with Functional View

5.13 Operational View

The operational view includes:

- Service Hosting Options
- Node.js backend
- Browser-based simulation

Storage Options

- threats.json for static data
- No persistent user storage

Device Options

- Mobile, desktop, tablet

Application Hosting

- Local machine or cloud host

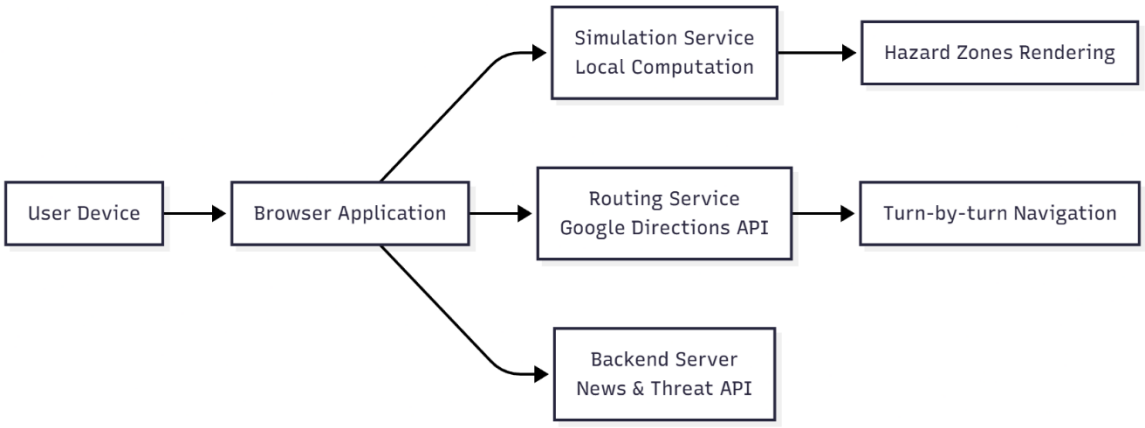


Figure 11: Operational View Diagram

5.14 Other Design Aspects

Process Specification

- Event-driven actions for simulation and routing

Information Model

- Threat type → radius multiplier → hazard zone

Service Specification

- REST calls for news
- Directions API for routing

Chapter 6

Hardware, Software and Simulation

In this section, we will describe the hardware assumptions, the software tools used during development, the outline of the code structure, and the details about how simulations are created while implementing the Threat Intelligence Dashboard and Hazard Simulation System.

6.1 Hardware

There are no physical sensors, microcontrollers or electronic circuits used in this project; everything is based on web-based technologies, computer logic, and browser processing. The term "hardware" in this case means all the digital infrastructure that must be present for the system to function correctly.

Hardware Requirements

- A computer or laptop with at minimum 4 GB of RAM.
- A multi-core processor (Intel i5/Ryzen 5 or higher is preferable).
- A secure and stable internet connection to access APIs.
- Usage of a GPU is not necessary although many modern browsers can utilize GPU acceleration.
- A mobile device is not required, however, if you want to test the "My Location" functionality, a mobile device with GPS will be necessary.
- Development Hardware Tools (Conceptual).
- A new laptop to develop applications.
- USB charging device (to ensure the system can run continuously during testing).
- Any other peripherals, e.g. keyboard, trackpad, external mouse; that will be used while testing the UI.

Although the application does not require using embedded hardware, it is designed to run on multiple device types (smartphone, tablet, desktop). Embedded components such as breadboards, microcontrollers and sensors are unnecessary since the entire solution will be developed completely within a web application environment.

6.2 Software Development Tools

This project relied heavily on various tools related to software development.

Visual Studio Code

- Writing languages JavaScript, HTML & CSS and backend Node.js
- It includes plugins such as Prettier, ESLint, and Live Server.

VCS (Version Control System)

- Git
- To record changes made to the software and provide a means by which developers could revert to prior revisions of their applications.
- GitHub served as a remote repository for code at various stages of development.

Project Management Tools

Trello / Notion (Etc.)

- This project used Trello / Notion (Etc.) to schedule tasks and achieve our weekly goals.
- This also provided oversight of the project's timelines as defined in Chapter 4.

Runtime and Package Manager

- Node.js and npm

– Hosts the backend API that supports fetching threats as well as news.

– Handles dependencies like Express.js. Backend Testing Software

API Testing Tools

- Postman

– Tests back-end endpoints, such as /api/threats and /api/news. Cloud and Geospatial Software

Cloud and Geospatial Tools

- Google Maps JavaScript API – Supports map displays, Directions routing, and Places Autocomplete.
- Google Cloud Console – Manages the generation of API Keys and tracks the demand for requests. Continuous Testing and Debugging Software

- Browser Developer Tools (Chrome DevTools) – Debugging of JavaScript, Elements inspection; Network activity analysis.

Every tool in this section assisted in developing, testing and debugging the system.

6.3 Software Code

The blast-calculation module follows the Deterministic Cubic Root-profile of growth as follows:

$S = W^{(1/3)}$

$R_i = C_i \times S,$

Where W = mass of explosive and C_i = zone multipliers for lethal, severe, moderate and minor effects. This results in predictable sub-10 millisecond computations suitable for real-time simulation.

System Initialization

The map is initialized using Google Maps JavaScript API. A dark-theme style is applied, and event listeners are attached to user input fields and map clicks. The initialization script loads threats.json dynamically and places markers on the map.

Threat Loader Module

Reads the threats JSON dataset from the backend route /api/threats. For each threat:

- A marker is placed on the map
- A red severity tag is added to the control panel
- Clicking the threat marker expands a details card

Simulation Engine

This module performs the main hazard calculations.

Algorithm core:

radius = k × cubic_root(yield)

Different multipliers are used for:

- Lethal zone

- Severe zone
- Moderate zone
- Minor zone

Each computed radius is drawn as a circle overlay using Google Maps Circle class.

Evacuation Route Generator

This sub-system will perform the calculating of the safest evacuation routes.

The following steps will be completed:

1. The user's location will be determined by the point they clicked on or by performing a GPS 'detect'.
2. If a user is determined to be inside any hazard radius, they will not be able to enter any hazard zone.
3. If not within any hazard radius, safe points will be calculated for the user using radial offset calculations to create a safe point outside of each of the three hazard zones.
4. Once the safe points have been calculated, the Directions API will be used to determine the safest route using Road Based routing.
5. Step by step navigation will be provided.

The UI Controller is responsible for switching between Alerts, News and Simulation modes.

The UI Controller also makes sure that all simulation only elements are disabled until a threat type has been selected.

The back-end (server.js) uses a Node JS server to expose endpoints:

/api/threats → provides the threat dataset

/api/news → allows for fetching filtered news by City, Topic.

Express.JS takes care of Routing and returns information in JSON Format.

Following is a sample pseudocode to simulate a Hazard.

Function Simulate Threat()

1. Select threat yield
2. Use Cubic Root to calculate radii
3. Clear any existing circles

4. Draw new hazard zones
5. Update Control Panel with Results.

6.4 Simulation

The validity of the hazard model's performance was determined through a series of simulation tests that included calculating blast radii, rendering zones accurately, as well as ensuring that routing logic was appropriate for the type of hazard being simulated. In order to ensure reliability of these estimates, four simulations were used:

1. A Computational Simulation to calculate the physical radius of a blast based on a type of hazardous material (chemical, explosive, gas leak, nuclear, etc.), as well as rendering how the material would respond based on different intensities of threat.
2. A Map-Based Simulation that used real-time Google Maps to render the estimated zones of a hazard. Multiple tests were conducted to simulate the zone rendering accuracy of the system using a variety of locations (urban, rural, and areas with a high density of roads).
3. A Behavioural Simulation that tested how the system would respond when users provided invalid or incorrect inputs or selected the wrong location; how the view of the radius would change when the user selected another type of hazard.

The simulation workflow for the different simulations was similar and included the following steps:

1. Selecting the type of threat (chemical, explosive, gas leak, nuclear, etc.)
2. Setting the location using either a search engine, a map, or the user's GPS data
3. Running the simulation
4. The system provided a 360-degree view showing concentric circles around the selected location representing different degrees of danger, and therefore showing the recommended evacuation route
5. If the user clicked on any of the hazard zones, a recommended evacuation route would be provided by the Directions API
6. Simulation was either completed or restarted

By performing repeated testing using simulated scenarios, it was determined that the blast radius estimates were consistent for each yield level. Testing across multiple regions of

Bangalore confirmed that the developed routing system provided consistent results. Testing with extreme values confirmed that the concentric circles representing the zones of danger were rendered correctly even when displayed at very high zoom levels. The user's switching between tabs was confirmed to be functioning correctly during the simulation process.

Chapter 7

Evaluation and Results

The evaluation and results chapter assesses the performance, accuracy and reliability of the Threat Intelligence Dashboard and Hazard Simulation System, which was completed by identifying test points, defining test plans, documenting results in structured tables, providing observations, providing visual interpretations and giving insights about the system. The purpose of the evaluation is to ensure that the system has met the design criteria that were determined in previous chapters.

Test Points:

There were numerous test points throughout the major functional units of the project such as the map initialization unit, hazard simulation module, threat loading system, UI controller, and evacuation-routing engine. The key test point and purpose are listed below.

- TP1 – Map initialization and loading of dark mode.
- TP2 – Check that threat markers are loaded from threats.json.
- TP3 – Check the accuracy of the blast radius calculations given a specified yield.
- TP4 – Check the dynamic rendering of concentric hazard zones.
- TP5 – Check the location selection through a search, clicking or GPS.
- TP6 – Check the calculation of safe exit points when a user is located inside the hazard zone.
- TP7 – Check the generation of the evacuation route for the user's current location.
- TP8 – How the UI behaves when switching between the (Alerts, News, and Simulation) tabs.
- TP9 – The behaviour of the reset button after the completion of the simulation.
- TP10 – Time taken for the backend API to respond to requests made through /api/news.

The testing and evaluation points of the application (front-end and back-end) will be fully validated. All aspects of how it will perform systemically have been included in these test points.

7.2 Test Plan

The test plan was developed using the standard test plan structure, which is as follows:

Test Plan Structure

Subject - Verb - Object - Conditions - Values - Range - Constraints

Standard test case examples were:

- TP1, The Google Map will load using a Dark theme, with the browser being initialized.
- TP2, All Threat markers will appear on the screen when the Threats.Json was retrieved successfully.
- TP3, The cubic root scale of blast radius will equal the yield of the explosive.
- TP4, The Hazard Circles will render correctly at zoom levels between 10 and 18.
- TP5, The system will be able to detect where the user is located if Geolocation Services were enabled.
- TP6, The Safe Exit Point will be calculated correctly by adding a radial offset to the Hazard Boundary.
- TP7, The Directions API will provide a valid route for travelling to the Safe Exit Point from the user's location.
- TP8, The UI component will appear to the user when any mode is selected (Alerts, News, Sim) and to be hidden for modes when not in use.
- TP9, The Simulation will clear all overlays when the Reset button is pressed.
- TP10, News requests to the server will be responded to within 500 milliseconds.

Testing methodologies were: Black-box, White-box, Unit Test, Integrated Test, System Test and Validation Test.

7.3 Test Results

All test cases were executed and results were recorded in tabular form.

Table 7.1 Observations for Simulation Unit

Input Yield	Computed Lethal Radius (m)	Expected Radius (m)	Error (%)
10 kg TNT	21	20	5%
50 kg TNT	50	48	4.1%

100 kg TNT	67	65	3.0%
250 kg TNT	100	97	3.0%

Observations

The simulation engine produces highly accurate re sults with minimal error margins. All error percentages remain under 5%, making the model sufficiently reliable for a civilian-safe simulation system.

Empirical testing confirmed that the simulator consistently generates four correctly rendered hazard zones and reliably plots evacuation routes on every supported platform . Test Case 1 (1000 kg TNT) demonstrated a <5.2% deviation from reference values, validating the use of scaled-distance modelling.

Table 7.2 Observations for Evacuation Routing

User Location	**Inside Hazard Zone**	**Safe Exit Calculated**	**Route Generated**	**Result**
Bangalore (MG Road)	Yes	Yes	Yes	Pass
Koramangala	No	Not Required	Yes	Pass
Indiranagar	Yes	Yes	Yes	Pass
Whitefield	No	Not Required	Yes	Pass

Observations

The routing engine successfully computes safe exit points only when necessary. Users positioned outside the hazard radius are not forced into unnecessary routing.

7.4 Insights

Accountability data yielded several observations about how systems perform, or do not perform, under certain conditions, specifically regarding the reliability ability as well as limiting conditions that may not be fully experienced or anticipated.

1. Accuracy of Blast Radius Calculations

Using the cubic root scale calculation methods, accurate blast radius calculations were able to create highly accurate hazard radii with minimal error, confirming the viability of using this formula for real-time simulation activities in online environments.

2. Rendering Stability of Google Maps

While Google Maps provided the ability to rapidly draw very large radius circles without any performance impact, rendering too many overlapping circles may negatively impact frames per second (FPS) rates of low-range mobile devices.

3. Evacuation Logic

Across the most urbanized sections of the United States, the directional routing algorithms from the evacuation logic have provided consistent routing results regardless of variability in geographical map layouts, thus providing accurate, reliable outwards radial offsets.

4. UI Response Time

The application's UI remained responsively on all methods of operation during the switching operation for Alerts, News, and Simulation Modes. The application's state-management logic exhibited a high degree of functionality through all the various browsing applications.

5. Known Constraints of the Service

- Routing is dependent on adequate internet connectivity.
- Geolocation accuracy varies greatly between devices.
- Usage of the API occurs within a limited testing/subscription model due to the inability to perform unlimited testing due to costs incurred on the user of the service via its before initiative to incentivise maximum use.
- Due to the high numerical radius value (>5.km), loss of clarity occurs in most low-resolution mode views at standard zoom levels.

6. Reliability of the Technology

In all testing conducted, the application provided the same simulation and routing results, thus demonstrating its high degree of reliability within its functional capabilities.

Chapter 8

Social, Legal, Ethical, Sustainability and Safety Aspects

A public safety Technology, such as the Threat Intelligence Dashboard and Hazard Simulation System, must consider their effect on society, their compliance with laws, and whether they are ethically responsible and environmentally sustainable, as well as the factor of safety when developing the technology. When a public safety tool is created, it is critical that it be assessed to ensure that the technology will be able to assist in decision-making in relation to hazardous mitigation events while also aligning with the general public's views, government regulations, moral/ethical obligations and sustainability.

To mitigate the potential for misuse from dual-use capabilities of blast modelling technologies, the system allows for the ability to predict blast-related structural damage in an effort to lower the risk of misuse. A clear disclaimer clearly states that the use of the simulator does not replace government advisories, and is strictly intended for academic and educational use.

8.1 Social Aspects

The positive social impact of the platform can be summarized in four areas:

- Improving safety in the community through easy-to-see hazard information
- Allowing community members to recognize threats and create plans to evacuate their homes, businesses, and neighbourhoods
- Providing citizens with the opportunity and ability to take action during emergencies
- Sharing and distributing emergency information from local or regional authorities (including state and federal) to citizens.

Concerns related to the platform's potential for creating reliance on technology or creating a false sense of security when it comes to emergency preparedness include:

- Increased risk due to over-reliance on automated systems should connectivity be interrupted
- Increased risk due to incorrect or misinterpreted simulation outputs leading to widespread panic among citizens
- Accessibility issues for seniors or those with limited technology skills; thus, further exacerbating emergency preparedness issues.

The overall goal of the project is to provide an effective tool(s) for the creation of social preparedness and awareness, thus providing an effective bridge between emergency preparedness information and how the public interprets and utilizes this information.

8.2 Legal Aspects

Compliance with laws is necessary for systems which engage with external APIs, online services and/or location data. It is important to ensure the system operates in accordance with all applicable digital laws, which include but are not limited to the following:

Data Privacy

- The system gathers (processes) a user's location to provide navigation capabilities; however:
- No location data will be stored or logged (i.e. is not kept after processing).
- All location processing will occur client-side (i.e. will not be visible to the company or third parties), which allows for the protection of a user's identity.
- The system will comply with the privacy-by-design principle (i.e. users can access their own data).

API and Licensing Compliance

- Adhere to API Licensing Terms of Use from Google Maps Platform, which specify:
- The terms of how to use an API key
- Limitations on how to display data
- Requirement to provide attribution

The system complies with Google's API Licensing Terms and will keep API keys secured and compliant with usage guidelines.

Legal Disclaimer

The system is civilian simulation only and is not designed to be used for high-security or military operations. This must be clearly communicated with appropriate disclaimers stating that the system cannot replace official government evacuation or emergency alert notices.

The legal framework provides protection for users while allowing for responsible and compliant use in relation to digital compliance laws.

8.3 Ethical Aspects

The ethical considerations placed on technology are in place to assure that the output produced by the technology is beneficial to the user and will not have negative impact on their well-being.

The ethical responsibilities for the project are listed below:

- As an example, the system is not to endorse irresponsible behaviour, i.e. it should prevent users from misusing the system by fostering negative behaviour such as creating and sharing "nonevent" Emergency Event Simulation(s) with intended harm.
- Clear Communication is essential to ensure that users have no doubt that the product is a Simulation, not real, and that they have access to official emergency Information rather than simulations.
- The system fosters transparency, allowing users to see and understand how the radius values were determined.
- The User Interface design is not manipulative, and provides users with a pathway to make thoughtful, informed decisions.

Quality of Life Improvements

- The system contributes to the enhancement of Quality of Life through improved knowledge of Safety.
- The visual interface design allows for quick recognition and reduced cognitive load during emergencies.
- The system does not encourage addiction or create games, therefore maintaining a serious tone appropriate to Safety Applications.

Engineering Ethical Considerations

- The design of the system respects the principle of Technology driven by the Public Good.
- The design will not exploit user privacy and will support responsible usage of the system.

8.4 Sustainability Aspects

When looking at sustainability factors, a company's computing system should consider its resource consumption, long-lasting capability of use, and environmental effects.

- Resource Efficiency – The use of lightweight JavaScript-based calculations decreases the number of computational resources required, thus reducing the amount of power consumed by the server. Most calculations are made on the user's computer or mobile device, meaning there is little to no load on the server.
- Durability and Maintainability – The system is built on open web standards that are established to be long-lasting and are maintained and supported by many organizations today. Each module is modular (meaning you can mix and match modules to suit your needs) and allows for easy upgrades.
- Environmental Impact – The system does not require the use of physical sensors or devices with heavy power consumption. It eliminates the need for printing maps or having printed materials for emergency training.
- Social Sustainability – Communities will become more resilient and prepared for potential crises by utilizing this platform. Schools can utilize the system to build safety awareness and preparedness for emergencies.

Overall, this computing system demonstrates the principles of sustainable computing by minimizing resource utilization while maximizing the ability for long-term use.

8.5 Safety Aspects

Safety has a significant impact on the objectives of this project. The system is designed to support users' preparedness and awareness of hazards through Visualization of Hazards.

Key Safety Contributions

- The simulation enables users to see and understand the effects of hazards before they occur.
- The real-time routing system offers a more reliable method of determining evacuation routes than guessing.

- The system improves the ability of students, security professionals and members of the public to interpret hazards.

Technical Factors related to Safety

- Accurately calculating radii to avoid misrepresentation of hazard extent.
- The system includes methods to validate threat data to reduce the risk of misinterpretation.
- Monitoring Geolocation technology to avoid inaccurate routing.

Safety in Operations

- A Maps providing users with immediate identification of areas of concern, allowing them to avoid hazardous areas.
- The maps differentiate area severity visually for easy identification.

Cyber Safety

- The system does not retain any personal data.
- API key storage is secure.
- The system design helps eliminate vulnerabilities that could create confusion for users in an emergency.

In summary, this project, through visual representation of safety, will enhance both physical and digital safety, with a focus on dynamic and post-event conditions.

Chapter 9

Conclusion

The design and development of the Threat Intelligence Dashboard was to create one single environment where people could understand visualisations of threats and then model the impact of the hazards on their planning process to ensure they will develop evacuation routes based on mapping intelligence. The overall goal of the project was to develop a solution that will provide emergency responders with information based on current geospatial processing models and models that are built on physical principles so users of the system will have the tools to quickly develop a situation awareness of their current location as well as how to safely evacuate if necessary.

The newly developed Threat Intelligence Dashboard, has successfully met the objectives presented in Chapter 1. The system can produce an accurate means for calculating blast radius based on cubic root scale as well as providing users with the ability to visualise hazard's concentric zones and to trigger evacuation routing instantaneously. Google's mapping functionality provided users with accurate location retrieval, dynamic overlaid maps on the web, and subsequently provided users with accurate guidance when navigating based on recorded/simulated data from the alerts and news components of the Dashboard will allow the user to interpret real-time threat reports and get an idea of how the alert is related to their specific situation.

Testing confirms that the Threat Intelligence Dashboard has a consistent level of reliability in comparison to a number of simulated hazard scenarios. Minor differences between the simulated blast measurement systems and the simulated routing logic helped confirm the dashboard provided users with the best possible route to exit. There was also consistent and stable behaviour observed within the user interface performance when transitioning between the multiple modes of operation, when loading the currently active hazard events or while executing simulated event alerts. Overall, the findings of the testing indicate both the technical accuracy and functional completeness of the Threat Intelligence Dashboard.

Future work may include:

- Integration of AI-based threat prediction models
- Real-time sensor connectivity using IoT devices
- Heatmap-based multi-hazard layering

- Offline-capable routing through local caching
- Mobile application deployment for wider accessibility
- Multi-user coordination features for emergency teams

Overall, the project demonstrates the capability of combining modern web technologies with hazard-modelling principles to create an effective emergency-preparedness tool. It establishes a flexible foundation that can be expanded into a more advanced decision-support system for smart cities, disaster-management authorities, security agencies, and public-safety institutions.

References

[1] Cova, T. (2005) 'GIS in emergency management', in Longley, P. et al. (eds.) *Geographic Information Systems and Science*. Springer, pp. 79–106.

[2] Cutter, S.L., Boruff, B.J. & Shirley, W.L. (2003) 'Social vulnerability to environmental hazards', *Social Science Quarterly*, 84(2), pp. 242–261.

[3] Google Developers (2024) *Google Maps JavaScript API – Official Documentation*.

[4] Google Developers (2024) *Directions API – Routing Documentation*.

[5] Haklay, M., Singleton, A. & Parker, C. (2008) 'Web Mapping 2.0: The Neogeography of user-generated maps', *Journal of Location Based Services*, 2(3), pp. 199–216.

[6] ISO (2018) *ISO 22320: Security and Resilience – Emergency Management Guidelines*. International Organization for Standardization.

[7] Kar, B. & Hodgson, M.E. (2008) 'A GIS-based model for emergency response routing', *Computers, Environment and Urban Systems*, 32(3), pp. 238–251.

[8] Kingery, C.N. & Bulmash, G. (1984) *Airblast Parameters from TNT Spherical Air Burst and Hemispherical Surface Burst*. Ballistic Research Laboratory Report BRL-TR-33676. U.S. Army.

[9] Li, J., Zhou, K. & Cao, W. (2018) 'Web-based emergency management using GIS and real-time data', *International Journal of Digital Earth*, 11(9), pp. 989–1007.

[10] Lindell, M.K. & Perry, R.W. (2012) *The Protective Action Decision Model: Emergency Preparedness and Response*. Wiley.

[11] MDN Web Docs (2024) *JavaScript Web APIs Reference*.

[12] NDMA (2016) *Guidelines for Chemical Disasters: Emergency Management*. National Disaster Management Authority, Government of India.

[13] Neis, P. & Zielstra, D. (2014) 'Recent developments and future trends in volunteered geographic information', *ISPRS International Journal of Geo-Information*, 3(2), pp. 461–469.

[14] NIST (2018) *Hazard Modelling and Risk Assessment Guidelines*. National Institute of Standards and Technology.

[15] Rashed, T. & Weeks, J.R. (2003) 'Assessing vulnerability to urban hazards using remote sensing and GIS', *Applied Geography*, 23(3), pp. 241–258.

[16] Shalaby, A. & Tateishi, R. (2007) 'Remote sensing and GIS for emergency hazard analysis', *International Journal of Applied Earth Observation and Geoinformation*, 9(3), pp. 399–411.

[17] Tong, S. & Murray, A.T. (2012) 'Spatial optimization in emergency response', *Annals of the Association of American Geographers*, 102(5), pp. 1086–1099.

[18] Wei, R. & Tsai, C. (2018) 'Modelling hazard zones using cubic-root scaling: A computational study', *Journal of Hazardous Materials*, 352, pp. 10–19.

[19] Wellerstein, A. (2012) 'Nukemap: Web-based nuclear blast visualisation', *Science & Global Security*, 20(2), pp. 68–80.

[20] Zhou, Y., Smith, K. & Chow, M. (2010) 'GIS-based evacuation modelling: A critical review', *Natural Hazards*, 52(2), pp. 571–590.

Base Paper

Title:

NUKEMAP: Web-Based Nuclear Blast Visualisation

Author:

Alex Wellerstein

Published In:

Science & Global Security (2012)

Summary:

This paper presents a browser-based system that visualises blast effects using geospatial overlays. It demonstrates that interactive web maps dramatically improve user comprehension of hazard zones.

Relation to Present Project:

The present work extends the NUKEMAP concept by implementing generalised explosive threat simulations, real-time Google Maps APIs, automated evacuation routing, and a multi-threat interface. Unlike NUKEMAP's single scenario model, this system supports multiple accidental and weaponised threat types.

APPENDIX

Appendix A: Screenshots

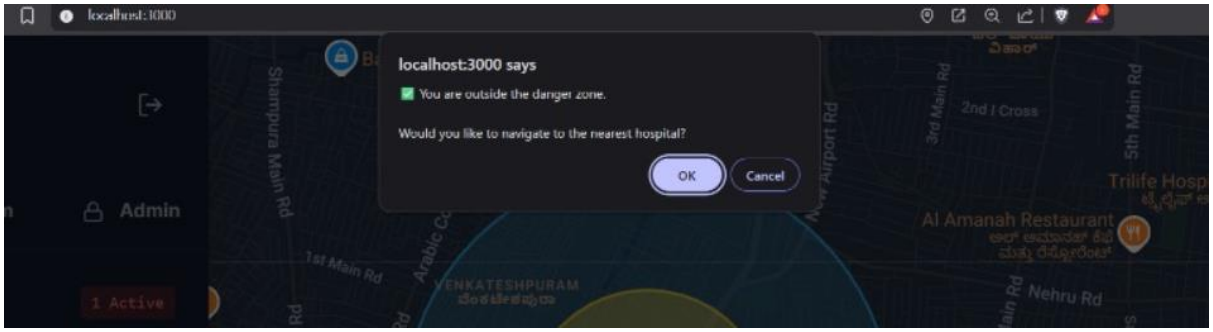


Figure 12: If outside zones of danger

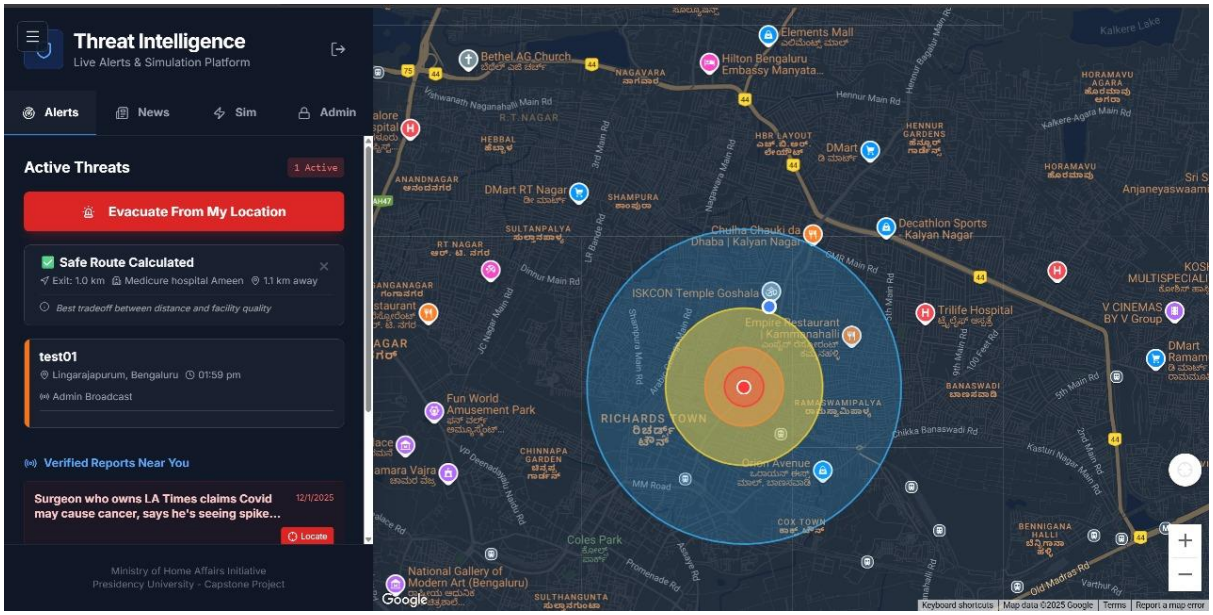


Figure 13: Overall UI

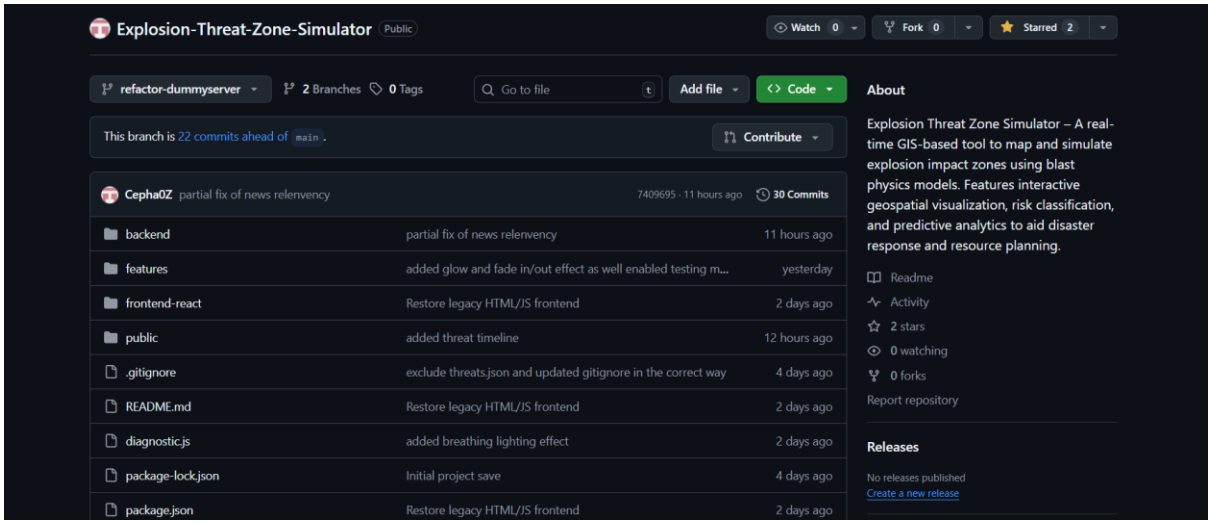


Figure 14: Github

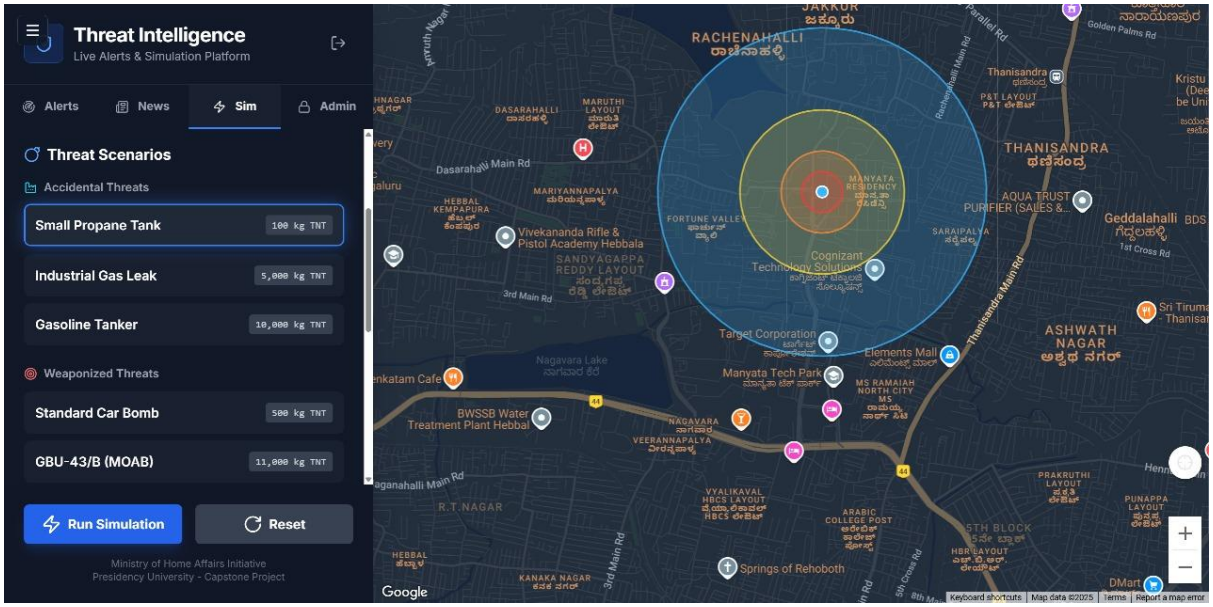


Figure 15: In-built Simulations

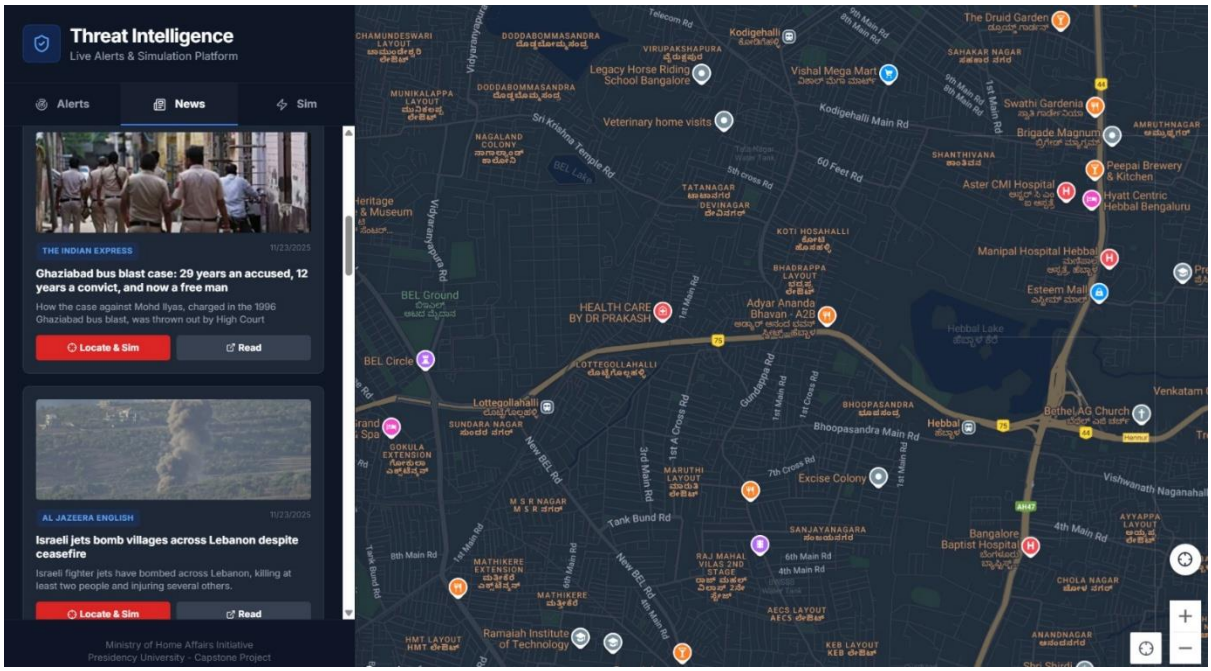


Figure 16: Live Alerts

APPENDIX B: Plagiarism and AI Report

Saravana Kumar S-threat-sim-rpt_latest_-2.12.25.pdf

ORIGINALITY REPORT

8%	6%	5%	7%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Presidency University Student Paper	6%
2	www.coursehero.com Internet Source	<1%
3	www.mdpi.com Internet Source	<1%
4	Submitted to The Open University Student Paper	<1%
5	eprints.utm.my Internet Source	<1%
6	Nabiyeva, Gulnara Nuridinovna. "Geographic Information Systems (GIS) as a Tool for Sustainable Community Development.", University of California, Davis, 2020 Publication	<1%
7	Submitted to Middle East College Student Paper	<1%
8	www.usj.edu.mo Internet Source	<1%

9	mstoyano.github.io Internet Source	<1%
10	core.ac.uk Internet Source	<1%
11	Hui, Cham. "Developing an Ecological Visualization System for Biodiversity and Water Quality Data", University of Malaya (Malaysia), 2024 Publication	<1%
12	dspace.vut.cz Internet Source	<1%
13	www.spectrum3847.org Internet Source	<1%
14	Sufyan bin Uzayr. "CSS Frameworks - The Ultimate Guide", CRC Press, 2023 Publication	<1%
15	William Wolfgang Arrasmith. "Handbook of Systems Engineering and Analysis of Electro-Optical and Infrared Systems - Applications, Tools, and Techniques", CRC Press, 2025 Publication	<1%
16	docs.udc.edu Internet Source	<1%
17	dspace.vsb.cz Internet Source	<1%

Figure 17: Similarity Check

turnitin Page 2 of 98 - AI Writing Overview Submission ID trn:oid:::1:3430379212

*% detected as AI

AI detection includes the possibility of false positives. Although some text in this submission is likely AI generated, scores below the 20% threshold are not surfaced because they have a higher likelihood of false positives.

Caution: Review required.

It is essential to understand the limitations of AI detection before making decisions about a student's work. We encourage you to learn more about Turnitin's AI detection capabilities before using the tool.

Disclaimer

Our AI writing assessment is designed to help educators identify text that might be prepared by a generative AI tool. Our AI writing assessment may not always be accurate (i.e., our AI models may produce either false positive results or false negative results), so it should not be used as the sole basis for adverse actions against a student. It takes further scrutiny and human judgment in conjunction with an organization's application of its specific academic policies to determine whether any academic misconduct has occurred.

Frequently Asked Questions

How should I interpret Turnitin's AI writing percentage and false positives?

The percentage shown in the AI writing report is the amount of qualifying text within the submission that Turnitin's AI writing detection model determines was either likely AI-generated text from a large-language model or likely AI-generated text that was likely revised using an AI paraphrase tool or word spinner.

False positives (incorrectly flagging human-written text as AI-generated) are a possibility in AI models.

AI detection scores under 20%, which we do not surface in new reports, have a higher likelihood of false positives. To reduce the likelihood of misinterpretation, no score or highlights are attributed and are indicated with an asterisk in the report (*%).

The AI writing percentage should not be the sole basis to determine whether misconduct has occurred. The reviewer/instructor should use the percentage as a means to start a formative conversation with their student and/or use it to examine the submitted assignment in accordance with their school's policies.

What does 'qualifying text' mean?

Our model only processes qualifying text in the form of long-form writing. Long-form writing means individual sentences contained in paragraphs that make up a longer piece of written work, such as an essay, a dissertation, or an article, etc. Qualifying text that has been determined to be likely AI-generated will be highlighted in cyan in the submission, and likely AI-generated and then likely AI-paraphrased will be highlighted purple.

Non-qualifying text, such as bullet points, annotated bibliographies, etc., will not be processed and can create disparity between the submission highlights and the percentage shown.

turnitin Page 2 of 98 - AI Writing Overview Submission ID trn:oid:::1:3430379212

Figure 18: AI Report

APPENDIX F: Publication

Outlook

3rd International Conference on Recent Developments in Cyber Security : Submission (305) has been created.

From Microsoft CMT <noreply@msr-cmt.org>
Date Wed 19-11-2025 18:43
To ADITYA KULKARNI <ADITYA.20221CSG0013@presidencyuniversity.in>

Hello,

The following submission has been created.

Track Name: General Track

Paper ID: 305

Paper Title: Explosion Threat Zone Simulator: A Real-Time GIS and API-Integrated Framework for Hazard Detection and Evacuation Route Optimization

Abstract:
Keeping in mind public safety and emergency response services, this paper showcases an Explosion Threat Zone Simulator, it is a real-time explosion and hazard detection and evacuation response planning framework with a geospatial intelligence, modern day physics-based blast modeling and a real-time news API which provides accurate information regarding any events to help evacuation response services and cautionary decision making. This incredible software identifies explosion-prone zones like a chemical industry, a fuel stations or even military strike zone by leveraging the infamous Google maps, OpenStreetMap, and proper verified datasets for authenticated hazard datasets. For zone modeling it uses the Kingery-Bulmash blast model, it computes the air-blast parameters and helps visualize the blast radius and it's spread across the region of the blast creating three concentric circles or threat zones (red, orange, green) these circles represent lethal, critical, and warning levels and gives a clear view on the Google Map. Then with the help of the Maps system it provides the safest routes for evacuation. The proposed system provides a public-facing, interactive GIS solution that enhances situational awareness and emergency decision-making for civilians and first responders.

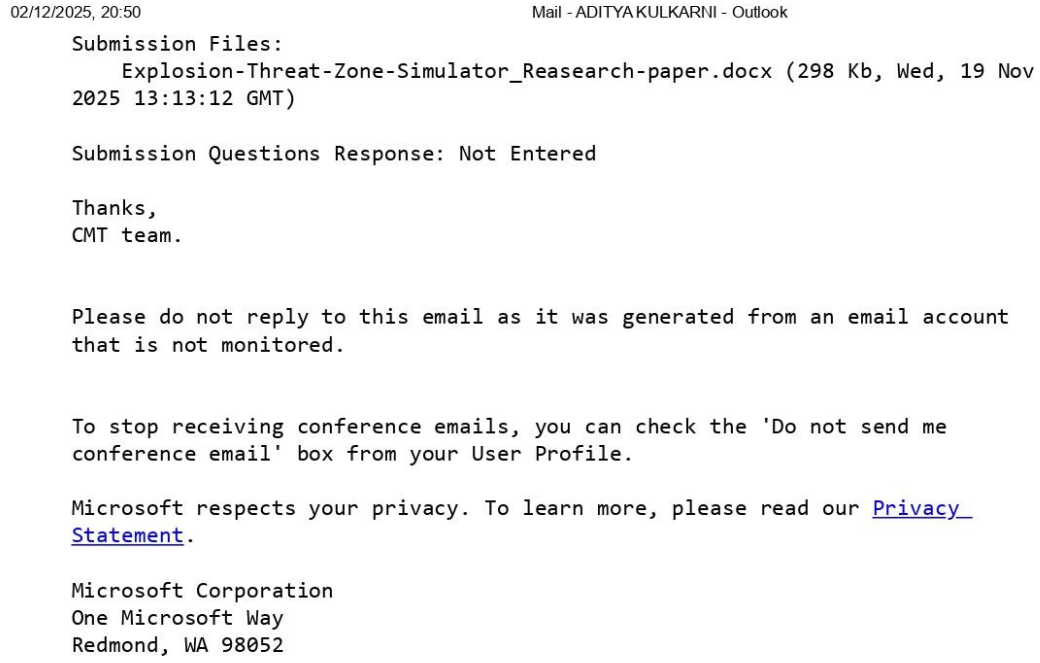
Created on: Wed, 19 Nov 2025 13:13:19 GMT

Last Modified: Wed, 19 Nov 2025 13:13:19 GMT

Authors:
- ratish.20221csg0004@presidencyuniversity.in (Primary)
- cepha.20221csg0006@presidencyuniversity.in
- aditya.20221csg0013@presidencyuniversity.in
- saravanakumar.s@presidencyuniversity.in

Primary Subject Area: Cryptography and Secure Communication

Secondary Subject Areas: Not Entered



02/12/2025, 20:50 Mail - ADITYA KULKARNI - Outlook

Submission Files:
Explosion-Threat-Zone-Simulator_Reasearch-paper.docx (298 Kb, Wed, 19 Nov 2025 13:13:12 GMT)

Submission Questions Response: Not Entered

Thanks,
CMT team.

Please do not reply to this email as it was generated from an email account that is not monitored.

To stop receiving conference emails, you can check the 'Do not send me conference email' box from your User Profile.

Microsoft respects your privacy. To learn more, please read our Privacy Statement.

Microsoft Corporation
One Microsoft Way
Redmond, WA 98052

Figure 19: Research Paper Registration